Demo: Snap – Rapid Sensornet Deployment with a Sensornet Appstore

Simon Duquennoy
simonduq@sics.se

Niklas Wirström
niwi@sics.se

Adam Dunkels
adam@sics.se

Swedish Institute of Computer Science
Box 1263, SE-16429 Kista, Sweden

Abstract

Despite ease of deployment being seen as a primary advantage of sensor networks, deployment remains difficult. We present Snap, a system for rapid sensornet deployment that allows sensor networks to be deployed, positioned, and reprogrammed through a sensornet appstore. Snap uses a smartphone interface that uses QR codes for node identification, a map interface for node positioning, and dynamic loading of applications on the nodes. Snap nodes run the Contiki operating system and its low-power IPv6 network stack that provides direct access from nodes to the smartphone application. We demonstrate rapid sensor node deployment, identification, positioning, and node reprogramming within seconds, over a multi-hop sensornet routing path with a WiFi-connected smartphone.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design

General Terms

Design

Keywords

Sensor Network, Deployment, Snap, Appstore

1 The Deployment Problem

Ease of deployment is often touted as a primary advantage of wireless sensor networks, yet sensor network deployment remains difficult. Existing sensor network systems are intended to serve a single purpose, where all nodes participate in serving this purpose. For example, typical TinyOS deployments run periodic data collection with the TinyOS Collection Tree Protocol (CTP) [4]. In CTP, one node (e.g. node with ID 0) is destined to be the sink node. Thus care must be taken at deployment [1]: the sink node must be given special attention, since it cannot be interchanged for other nodes. If the designated sink node would break during deployment, it cannot be easily replaced with another node.

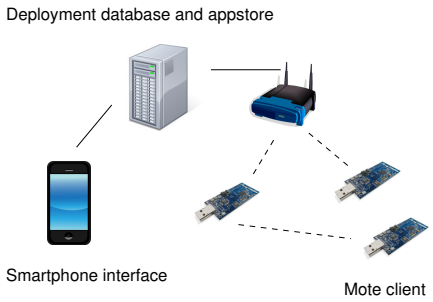

Figure 1. Snap consist of a mote client, a deployment database and appstore, and a smartphone interface.

In many emerging sensornet applications, the sensors are heterogeneous. For example, in a building automation system, nodes with light sensors can be placed in windows to determine the available sunlight [7] or near appliances to monitor their usage [8]. Both applications can make efficient use of the same hardware, thereby invoking economics of scale in the manufacturing of the devices, though their application is very different. Existing sensor network deployment methods do not provide any support for such applications.

2 The Snap Deployment System

Snap consists of three parts, as shown in Figure 1: a mote client, a deployment database and sensornet appstore, and a smartphone interface. The mote client communicates with the smartphone interface to receive configuration that the user sets on the smartphone. The appstore transmits loadable applications onto the motes over a low-power IPv6 network.

2.1 The Mote Client

The motes run the Contiki operating system and leverage its dynamic software loading functionality, its low-power IPv6 networking, RPL routing [5], and CoAP application protocol [6]. The mote client communicates with both the appstore and the smartphone interface.

To enable rapid software download, the mote software uses the burst forwarding technique, published at ACM SenSys 2011 [2], to provide a favorable trade-off between low-power operation and high throughput, even with lossy links. The software download is currently implemented through the block transfer mode of the CoAP application protocol [3].

SenSys'11, November 1–4, 2011, Seattle, WA, USA.
ACM 978-1-4503-0718-5/11/11

2.2 The Deployment Database and Appstore

The deployment database keeps track of the identities, positions, and installed software on each node, all of which are registered by the smartphone interface during node deployment. The appstore maintains binary versions of sensornet applications. The smartphone interface requests a list of applications from the appstore and presents it to the user. After the user has selected the application to load, the mote client requests the appropriate application from the appstore. The mote client includes information about the mote hardware platform so that the appstore can provide the correct binary version of the application. Both the deployment database and the appstore run on a server outside of the sensor network.

2.3 The Smartphone Interface

Snap uses a smartphone application for node identification, positioning, and reprogramming. Deployment is a four-step procedure: node identification and positioning, node registration, node configuration, and application installation. The user is involved only in the first and third steps. Snap processes steps two and four in the background.

1. **Node identification and positioning.** The user places the node at the physical location that the node is intended to sense at, and identifies the node by pointing the phone's camera at the QR code of the node (Figure 2). After the node has been identified, the user gives the node a position on a map by using the map interface on the phone.
2. **Node registration.** The smartphone application registers the node with the deployment database. Although this step is not strictly necessary for the operation of the network and its applications, the registration helps manual maintenance after deployment.
3. **Node configuration.** The user selects one or more applications to be installed on the node, as well as application configuration parameters. Snap sends the application selection along with the configuration parameters to the mote.
4. **Software installation.** The node downloads the applications from the application store and installs them on the node. The actual installation consists in dynamic linking of the downloaded Contiki process to the software initially deployed on the mote. The new application starts without requiring the mote to reboot, thus having little side effect on other running applications.

3 Demonstration

We demonstrate the full Snap system with a set of 10 Tmote Sky motes, an Android smartphone, and a deployment database and appstore, which runs on a laptop. Users can identify and position nodes in the smartphone interface and select software to be downloaded to the motes. One of the applications is a simple data collection application that reports data to a laptop computer, which shows the data being collected. Every mote runs a background application that estimates their power consumption with the Contiki powertrace power tracking mechanism and sends the data to the sink. Another laptop displays the power samples, which demonstrates the low-power operation of the system. We use

Figure 2. Node identification (left): the QR code on the mote is automatically identified and the containing information is decoded by the smartphone application. Node positioning (right): after the node has been identified, the user gives the position and direction of the node on the map view.

a WiFi router, configured with local IPv6 access as shown in Figure 1, for communication with the smartphone and the low-power IPv6 network.

The demo shows how Snap makes sensor network deployment easy, while maintaining a low power consumption within the sensor network, even in the face of heterogeneity and rapid system reconfiguration.

4 References

[1] M. Ceriotti, L. Mottola, G. P. Picco, A. Murphy, S. Guna, M. Corra, M. Pozzi, D. Zonta, and P. Zanon. Monitoring heritage buildings with wireless sensor networks: The torre aquila deployment. In *Proceedings of the International Conference on Information Processing in Sensor Networks (ACM/IEEE IPSN)*, Washington, DC, USA, 2009.

[2] S. Duquennoy, F. Österlind, and A. Dunkels. Lossy Links, Low Power, High Throughput. In *Proceedings of the International Conference on Embedded Networked Sensor Systems (ACM SenSys)*, Seattle, WA, USA, November 2011.

[3] S. Duquennoy, N. Wirström, N. Tsiftes, and A. Dunkels. Leveraging IP for Sensor Network Deployment. In *Proceedings of the workshop on Extending the Internet to Low power and Lossy Networks (IP+SN 2011)*, Chicago, IL, USA, April 2011.

[4] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis. Collection tree protocol. In *Proceedings of the International Conference on Embedded Networked Sensor Systems (ACM SenSys)*, Berkeley, CA, USA, 2009.

[5] J. Ko, J. Eriksson, N. Tsiftes, S. Dawson-Haggerty, M. Durvy, J. Vasseur, A. Terzis, A. Dunkels, and D. Culler. Beyond Interoperability: Pushing the Performance of Sensornet IP Stacks. In *Proceedings of the International Conference on Embedded Networked Sensor Systems (ACM SenSys)*, Seattle, WA, USA, November 2011.

[6] Matthias Kovatsch, Simon Duquennoy, and Adam Dunkels. A Low-Power CoAP for Contiki. In *Proceedings of the Workshop on Internet of Things Technology and Architectures (IEEE IoTech 2011)*, Valencia, Spain, October 2011.

[7] J. Lu, D. Birru, and K. Whitehouse. Using simple light sensors to achieve smart daylight harvesting. In *Proceedings of the 2nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*, Zurich, Switzerland, 2010.

[8] Z. Taysi, M. Guvensan, and T. Melodia. Tinyears: spying on house appliances with audio sensor nodes. In *Proceedings of the 2nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*, Zurich, Switzerland, 2010.